(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) Publication number: **0 217 486**
A2

(12) # EUROPEAN PATENT APPLICATION

(21) Application number: 86304507.6

(22) Date of filing: 12.06.86

(51) Int. Cl.⁴: G 06 F 13/42
H 04 L 7/00, G 06 F 5/06

(30) Priority: 27.09.85 US 781868

(43) Date of publication of application:
08.04.87 Bulletin 87/15

(84) Designated Contracting States:
DE FR GB IT

(71) Applicant: Hewlett-Packard Company
P.O. Box 10301
Palo Alto California 94303-0890(US)

(72) Inventor: Nordby, Rasmus
Kammerrabensvej 25 S.V.
DK-2970 Horsholm(DK)

(72) Inventor: Crandall, Douglas
45 Newell Road No. 220
E. Palo Alto California 94303(US)

(74) Representative: Colgan, Stephen James et al,
CARPMAELS & RANSFORD 43 Bloomsbury Square
London WC1A 2RA (GB)

(54) **A synchronizing system.**

(57) A synchronizing system is provided for reliably passing data across a boundary between two independent, non-correlated clocks, referred to as the receiving and transmitting clocks. The system reduces occurrence of errors due to asynchronous samplings to an arbitrarily low level based on metastable operation. The system is organized as a two port memory with unit distance code addressing the memory cells. It performs a handshake between the two non-correlated clock systems to allow for any ratio between the two clocks.

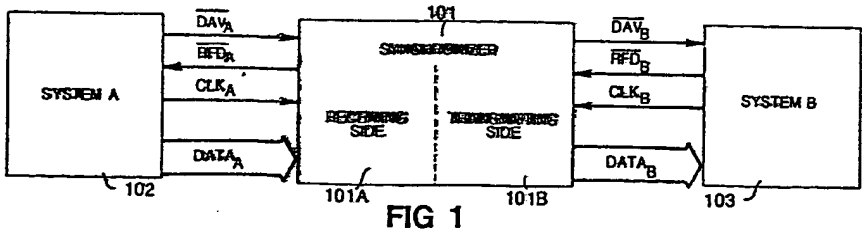


FIG 1

EP 0 217 486 A2

A SYNCHRONIZING SYSTEM

0217486

Background of the Invention

The present invention relates to a synchronizing system being a computer storage device, and more particularly relates to means used between two devices transmitting data at different speeds to counteract this differential as a buffer.

In any circuit interfacing two non-correlated and asynchronous clocks, there will exist a non-zero probability of sampling a signal from one system with the clock from the other system while the signal is changing. The result of such a sampling, where set-up and hold times are not met, may be an erroneous output. Unpredictably the output can be logic "1", logic "0" or in a "metastable state" between "1" and "0". This undefined logic state may be interpreted differently by different receiving devices, and in a general configuration it is able to propagate through the succeeding logic gates.

This "metastable state" is discussed in the articles, "General Theory of Metastable Operation" by Leonard R. Marino, P107-P115, IEEE TRANSACTIONS ON COMPUTERS, Vol. C.30, NO.2, February 1981, "Anomalous Behavior of Synchronizer and Arbiter Circuits" by Thomas J. Chaney et al, P421-P422, IEEE TRANSACTIONS ON COMPUTERS, Correspondence, April 1973, and "Circuit Technology in a large computer system" by D. J. Kinniment et al, P435-P441, THE RADIO AND ELECTRONIC ENGINEER, Vol. 43, No.7, July 1973. The metastable state is potentially unstable and the probability that the output will stay in metastable state decays exponentially

with time. All logic families exhibit the metastable operation, when sampling occurs on a transition. However, families with a high-gain bandwidth product are less inclined to exhibit the metastable state than those with a low-gain bandwidth product. The metastable problem is a fundamental problem in digital circuits interfacing asynchronous clocks, and there exists no exact solution to avoid errors. However, the probability of inducing errors due to the metastable operation can be reduced by allowing longer time for the output to settle from the metastable state and by employing high-gain bandwidth product devices.

Summary of the Invention

In accordance with a preferred embodiment of the present invention, there is provided a synchronizing system. The purpose of the synchronizing system is to reliably pass data across a boundary between two independent, non-correlated clocks; referred to as the receiving and transmitting clocks. Thus, the task is to minimize the occurrence of errors caused by asynchronous samplings. Data is being supplied to the synchronizer synchronously with the receiving clock and is transmitted to the receiver with a minimal delay from the synchronizer synchronously with the transmitting clock.

This synchronizer transfers data bytes sourced at one clock rate to a data sink which removes them at another clock rate. The two clock frequencies can have arbitrary phase and frequency relationship, i.e. asynchronous, faster or slower. It also performs a handshake at the input and output sides synchronous

with the input clock and output clock respectively.

The synchronizer made by the NMOS circuit consisting of a RAM with address logic allows for independent read and write access to the RAM. The circuit is optimized to minimize the error rate when the read and the write clocks are asynchronous. Error free transfer through the synchronizer occurs whenever metastable states generated by asynchronous samplings have settled to an arbitrary valid logic level, i.e. one or zero, within one clock cycle. The read and write clocks can be synchronous or asynchronous and they can be periodic or aperiodic. Furthermore, a low level synchronous handshake is implemented on each side of the synchronizer and allows for use of fill characters when no data is available and for data source hold-off when data overflow would otherwise occur. Independent enable lines on sink and source side can be used to place the outputs on each side in a high impedance state.

Brief Description of the Drawings

Figure 1 is a system block diagram of a preferred embodiment of the present invention describing the system configuration of a system A, a system B, and a synchronizer.

Figures 2A-B show a functional block diagram of the synchronizer employing a two port memory with address logic circuit.

Figures 3A-D show a circuit diagram of the synchronizer describing the detailed circuit configuration of the functional block diagram shown in Figures 2A-B.

Figure 4 is an illustration of the metastable state in a flip-flop used for the synchronizer.

Figure 5 is an illustration of the conventions used in the following timing charts.

Figure 6 is a timing chart of the receiving side of the synchronizing system.

Figure 7 is a timing chart of the transmitting side of the synchronizing system.

Figures 8A-C show a timing chart of the synchronizing system in cases where clock signal A is two times slower than clock signal B.

Figures 9A-C show a timing chart of the synchronizing system in cases where clock signal A is two times faster than clock signal B.

Figures 10A-C show a timing chart of the synchronizing system in cases where clock signal A is the same as clock signal B.

Description of the Preferred Embodiment

In Figure 1, a synchronizing system is shown as a system configuration. In the embodiment of the invention, the synchronizing system consists of a synchronizer 101 having a receiving side 101A coupled to a system A 102 and a transmitting side 101B coupled to a system B 103. The coupling of the synchronizer 101 to the system A 102 and the system B 103 is performed by a synchronous and interlocked handshake. It is synchronous because all lines in the interface are synchronous

with the transfer clock frequency. It is used on the input and output side of the synchronizer circuit in which case the synchronizer translates one synchronous handshake into a similar handshake synchronous with an arbitrary second clock. Data (DATA) is transferred in bytes of any width at the rate of a clock signal (CLK). Each data byte is labelled with a "data available" signal (DAV). For each clock cycle the data sink presents a "ready for data" signal (RFD).

In Figure 2, the synchronizer 101 is explained in more detail as a functional block diagram. The synchronizer 101 is organized as a two port memory 201 with a write address controlled by the receiving clock signal CLK A and the handshake lines on the receiving side and a read address controlled by the transmitting clock CLK B and the handshake lines on the transmitting side. By addressing the two port memory cells in a unit distance code which results in only one bit changing each time it advances to the next state, an important feature for asynchronous operation is obtained. Namely, the address control logic can be designed for asynchronous operation so that the outcome of the asynchronous sampling is no longer important. The only requirement for error free transfer is that the asynchronous sampling has settled to a valid logic one or zero level in whatever time the control logic allows for metastable settling. With 6 bits depth of the two port memory, one clock period is allowed for metastable settling. If for a given logic family this introduces a non-satisfactory error rate, the circuit architecture allows for arbitrarily long metastable settling

time, i.e. arbitrarily low error rate, with the expense of added depth to the two port memory and cascaded asynchronous samplings. Cascading M extra asynchronous samplings increases the through-delay by M transmitting clock cycles.

The function of the address control logic shown in Figure 2 is to determine the following three logic operations: first, whether a valid read out can be done from the two port memory and a write cycle of valid data has been completed or not; second, whether an excessive amount of data is accumulated in the two port memory and whether the next byte sent into the synchronizer can be accommodated in the two port memory or not; and third, whether a data overflow has actually occurred in the two port memory or not, in case the data source does not obey the handshake by a data overflow signal (DOV). The read and write operation will be described in more details hereafter. Writing data into the two port memory cell to which the write address is pointing is done each transmit clock cycle, if the synchronizer status is ready for data, and if valid data is available from the source. The synchronizer is considered to be ready for data when there is still one vacant cell left in the two port memory. To determine this vacancy condition, an asynchronous sampling of the read address with the receiving clock is required. The write address is advanced by one only after the completion of writing in; otherwise it is not advanced. Reading data out of the two port memory cell to which the read address is pointing is done each transmitting clock cycle, if the write operation into this cell has been completed. The write operation is considered

completed if the read address does not equal the write address. This comparison involves an asynchronous sampling of the write address with the receiving clock. The read address is advanced by one only after the completion of valid reading out; otherwise it is not advanced.

Detailed design of the synchronizer system shown in the functional block diagram of Figure 2 is disclosed in Figure 3, where the two port memory is, for example, an integrated circuit TYPE SN54170 manufactured by Texas Instruments Incorporated, and the unit distance code is a Gray code.

The synchronizing system will hereafter be more precisely explained as regards the timing of the operation. In Figure 4, a flip-flop 401 used for the synchronizer is shown, where metastable behavior of the output of the flip-flop 401 is illustrated. When the flip-flop 401 receives data and a clock signal, the output of the flip-flop 401 remains at an invalid level for an arbitrarily long time as illustrated in Figure 4 as part 402. When it finally resolves it may go to either high or low state, regardless of input, as illustrated in Figure 4 as part 403.

In Figure 5, abbreviations and conventions used for the following timing charts are shown.

In Figure 6, the receiving side of the synchronizer receives the clock signal CLK A, the data available signal DAV A, and the data DATA A including D1 and D2 from the system A. When the synchronizer is ready for receiving the data DATA A, it provides the ready for data signal RFD A to the system A. At the time T1

the synchronizer is not ready, so the data is held at interface and the data is not received by the synchronizer. At the time T2 the synchronizer is now ready, then the data D1 is received by the synchronizer. At the time T3 the synchronizer is still ready, so the data D2 is received by the synchronizer. At the time T4 the synchronizer is still ready, but the system A is not available, therefore the data is not received by the synchronizer.

In Figure 7, the transmitting side of the synchronizer receives the clock signal CLK B and the ready-for-data signal RFD B from the system B. When the synchronizer is available to transmit the data, it provides the data available signal DAV B and the data DATA B to the system B. At the time T1 the system B is not ready, so the synchronizer holds the data constant and the data is not transmitted by the synchronizer. At the time T2 the system B becomes ready, so the data D1 is transmitted to the system B. At the time T3 the system B is still ready, so the data D2 is transmitted to the system B. At the time T4 there is no data in the synchronizer, so no transmission occurs.

In referring to Figures 8 through 10, there may be three cases since the two interfacing clocks are non-correlated. First, Figure 8 shows that the receiving clock CLK A of the synchronizer system from the system A is slower than the transmitting clock CLK B of the synchronizer system to the system B, namely, the data is sent out of the synchronizer faster than it can be supplied. Second, Figure 9 shows that the receiving clock CLK A from the system A is faster than the transmitting

clock CLK B to the system B, namely, the data would potentially be supplied to the synchronizer faster than it would be taken out. Third, Figure 10 shows that the receiving clock CLK A from the system A is equal to the transmitting clock CLK B to the system B. In each of these cases the precise details of timing will depend on clock phases and frequencies, so the timing diagrams are representative but not exhaustive. In referring to Figure 8, when the data is removed from the synchronizer faster than it is supplied, valid data cannot be sent for each transmitting clock cycle. A handshake line indicates that no data is available, and the value of the data output should be ignored. If constant data is required on the output side, for example, in a communication system, this handshake line can be used to control the generation of idle or dummy characters. However, the synchronizer only indicates the presence or absence of valid data. It does not generate dummy characters in its own logic circuit. In referring to Figure 9, when data is supplied to the synchronizer faster than it is taken out, data will accumulate in the synchronizer. To avoid data overflow a handshake line is provided to the data source telling if too much data is being accumulated in the synchronizer. In referring to Figure 10, when data is supplied to the synchronizer at the same rate as it is being removed, the entirely synchronous operation through the systems can be performed. After a steady state filling of the two port memory is reached to accommodate the given phase difference between the receiving clock and the transmitting clock, the synchronizer will always be ready for

data and it will always have valid data available. Similarly, the synchronizer is capable of receiving information from the data source telling if valid data is sent, and from the sink telling if the sink is ready to accept more data. The operation of the synchronizer is not dependent on the continuity of the clocks, thus it will work with interrupted or varying clock rate as well. The maximum delay from write to read, that is, delay through the synchronizer, is the sum of one receiving clock cycl and one transmitting clock cycle. If N bytes of data were already accumulated in the synchronizer, an additional delay of N transmitting clock cycles are added.

As explained above, the essence of the synchronizing system is that the memory cells in the two port memory are addressed in a unit distance code so that sampling an address during a transition will give as an output either the previous address or the coming address, provided metastable setting is reached in th allowed time period. As a result, in an ambiguous sampling whic settles from the metastable state arbitrarily into a valid logic one or zero, the least desirable possibility would be a delay in writing one receiving clock cycle or in reading one transmitting clock cycle. This delaying effect does not accumulate after several consecutively ambiguous samplings based on the unit distance code. It is noted that data is never sampled asynchronously and that the outcome of the asynchronous address sampling is allowed to arbitrarily settle to a logic one or zero state. It is also an important feature of the synchronizing system that the synchronizer is capable of receiving the data

from the system A at a first address whose succeeding addresses are encoded in the unit distance code by determining that the two port memory is not full, and is also capable of transmitting the data to the system B from the same address to the first address and the succeeding addresses of the first address by determining that the two port memory is not empty.

We claim:

1. A synchronizing system comprising:

first system means having data synchronized with a first clock signal;

second system means having the data to be synchronized with a second clock signal being independent from the first clock signal; and

synchronizer means coupled to the first and second system means for transferring the data from the first system means to the second system means, the synchronizer means having a two port memory which receives the data from the first system means at a first address whose succeeding addresses are encoded in a unit distance code by determining that the two port memory is not full, and transmits the data to the second system means from the same address to the first address and the succeeding addresses by determining that the two port memory is not empty.

2. A synchronizing system according to claim 1, wherein the synchronizer means consists of a one chip integrated circuit.

3. A synchronizing system according to claim 1, wherein the first clock signal is faster than the second clock signal.

4. A synchronizing system according to claim 1, wherein the first clock signal is the same as the second clock signal.

5. A synchronizing system according to claim 1, wherein the first clock signal is slower than the second clock signal.

6. A synchronizing system according to claim 1, wherein the unit distance code is a Gray code.

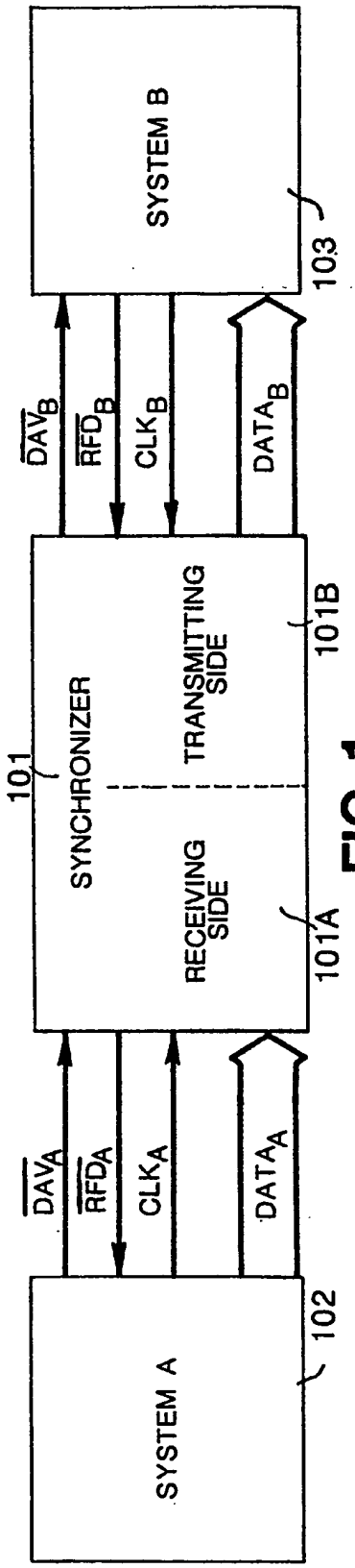


FIG 1

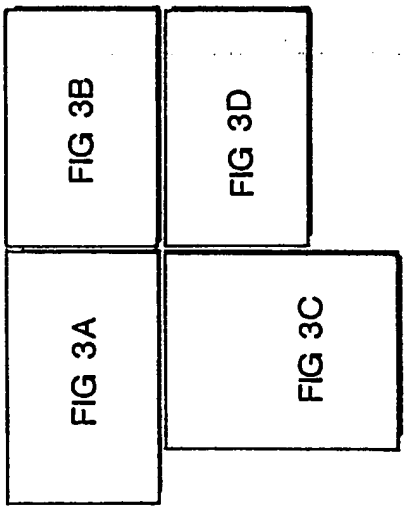


FIG 3

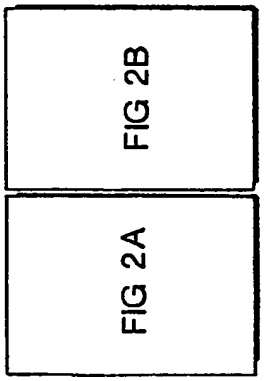


FIG 2

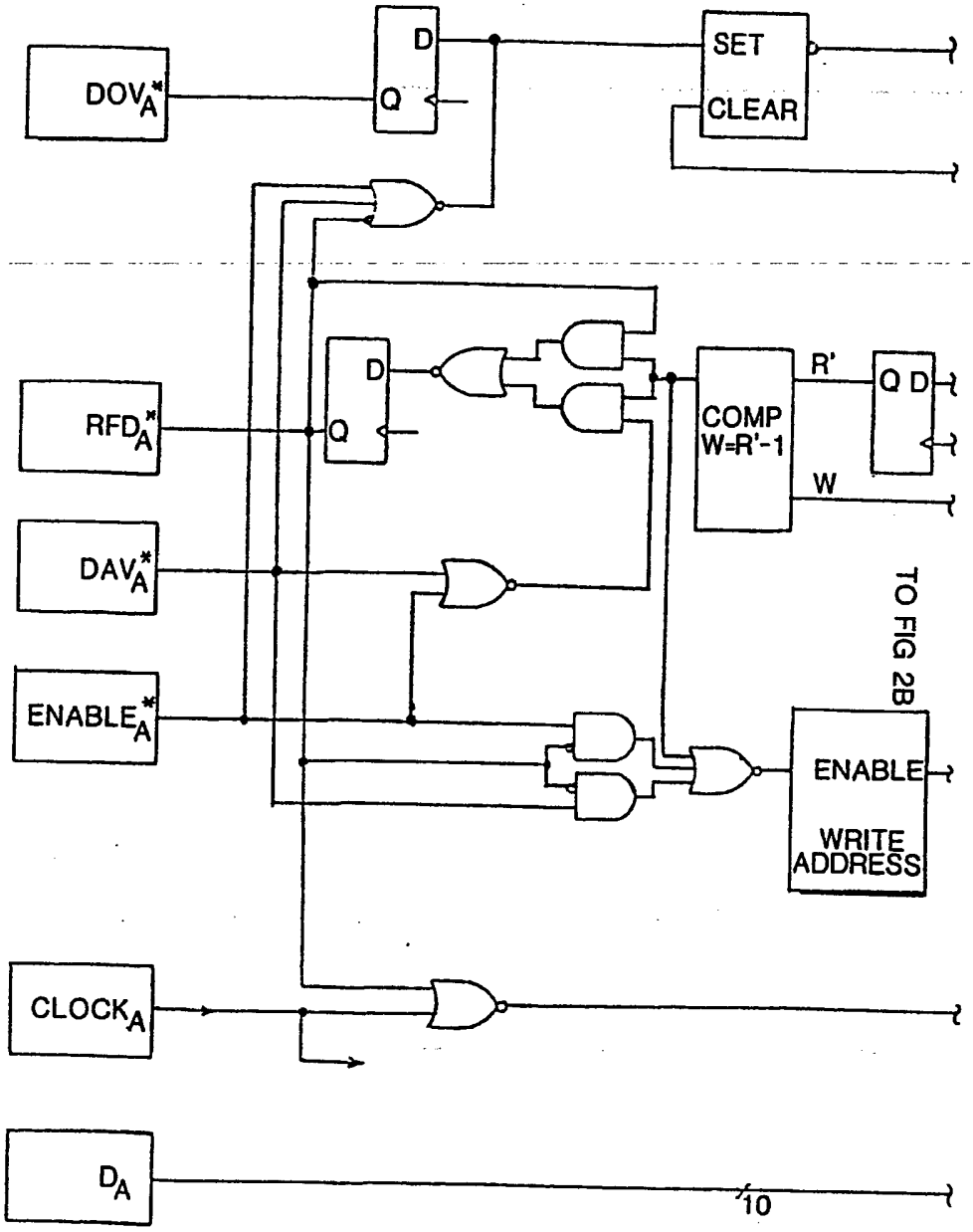

DOV*A
D
Q
SET
CLEAR
RFD*A
D
Q
COMP
W=R'-1
R'
Q D
W
DAV*A
ENABLE*A
TO FIG 2B
ENABLE
WRITE
ADDRESS
CLOCKA
DA
10

FIG 2A

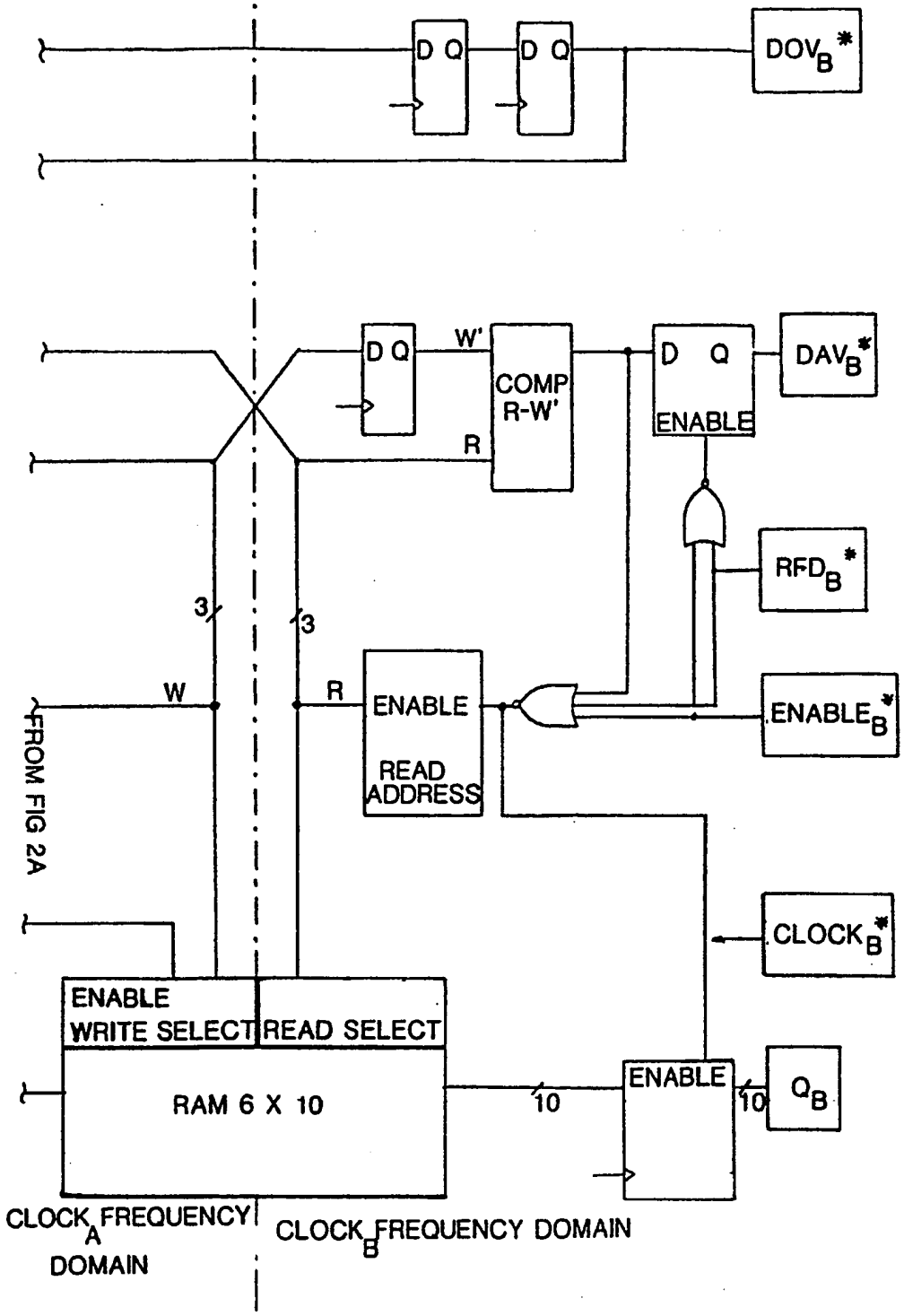

D Q
D Q
DOV$_B$*
D Q
W'
COMP
R-W'
R
D Q
ENABLE
DAV$_B$*
RFD$_B$*
3
3
W
R
ENABLE
READ
ADDRESS
ENABLE$_B$*
FROM FIG 2A
CLOCK$_B$*
ENABLE
WRITE SELECT
READ SELECT
RAM 6 X 10
10
ENABLE
10
Q$_B$
CLOCK$_A$ FREQUENCY
DOMAIN
CLOCK$_B$ FREQUENCY DOMAIN

FIG 2B

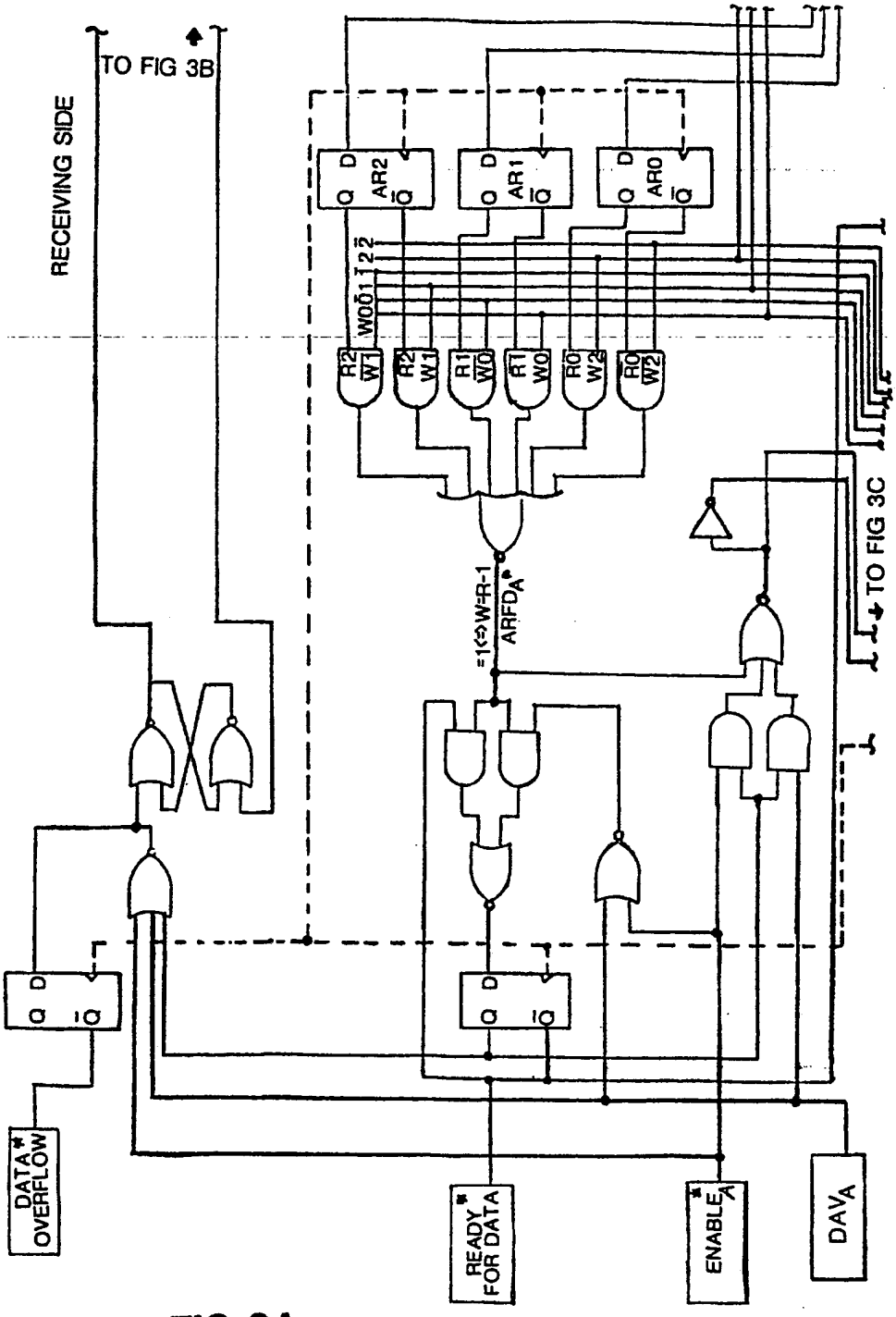
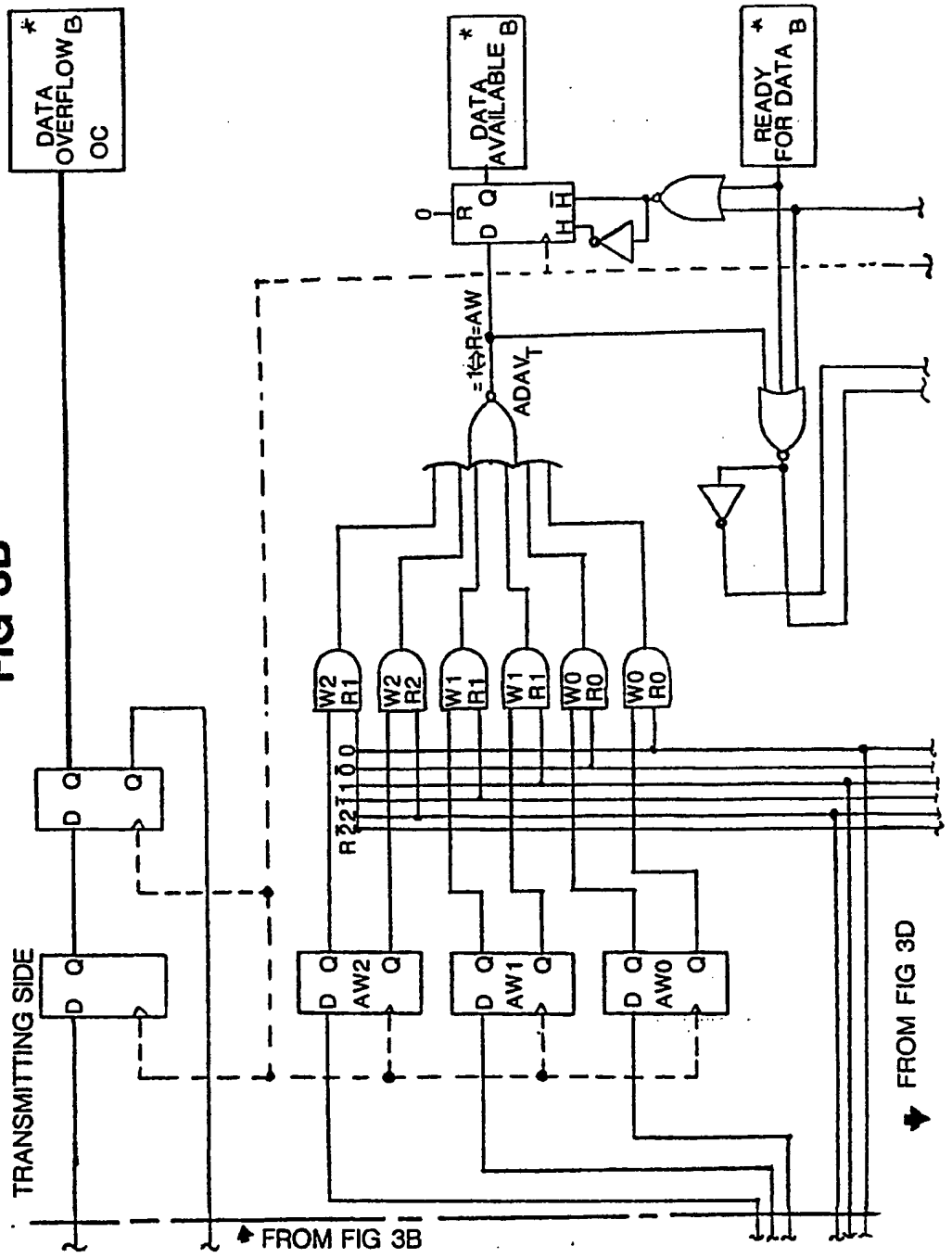

TO FIG 3B
RECEIVING SIDE
AR2
AR1
AR0
R2 W1
R2 W1
R1 W0
R1 W0
R0 W2
R0 W2
=1⇔W=R-1
ARFD*A
TO FIG 3C
DATA* OVERFLOW
READY* FOR DATA
ENABLE* A
DAVA

FIG 3A

FIG 3B


DATA OVERFLOW B
OC
DATA AVAILABLE B
READY FOR DATA B
R
D
Q
H
=1⇔R=AW
ADAV T
W2 R1
W2 R2
W1 R1
W1 R1
W0 R0
W0 R0
R2 R1 R0
AW2
AW1
AW0
TRANSMITTING SIDE
FROM FIG 3B
FROM FIG 3D

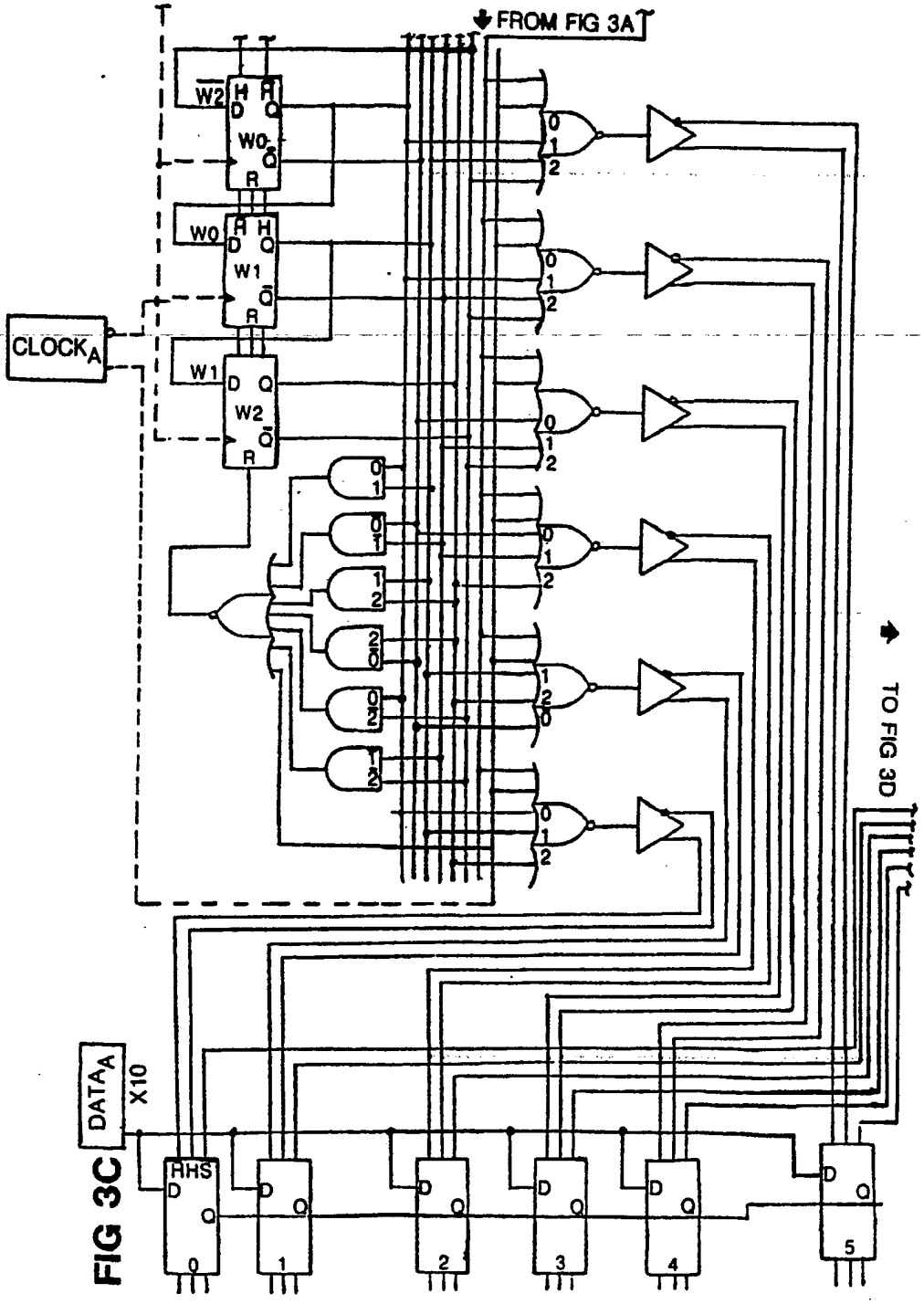


FIG 3C

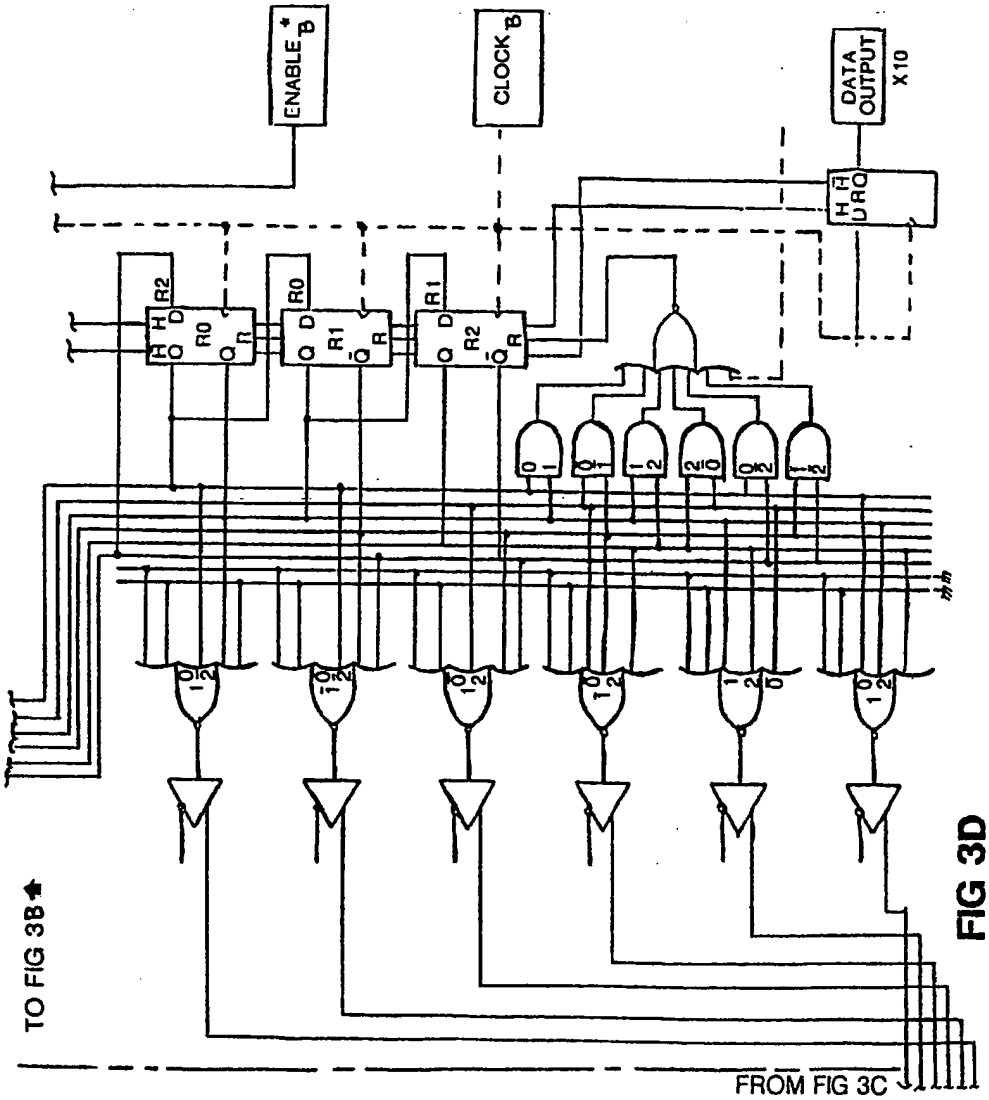


FIG 3D

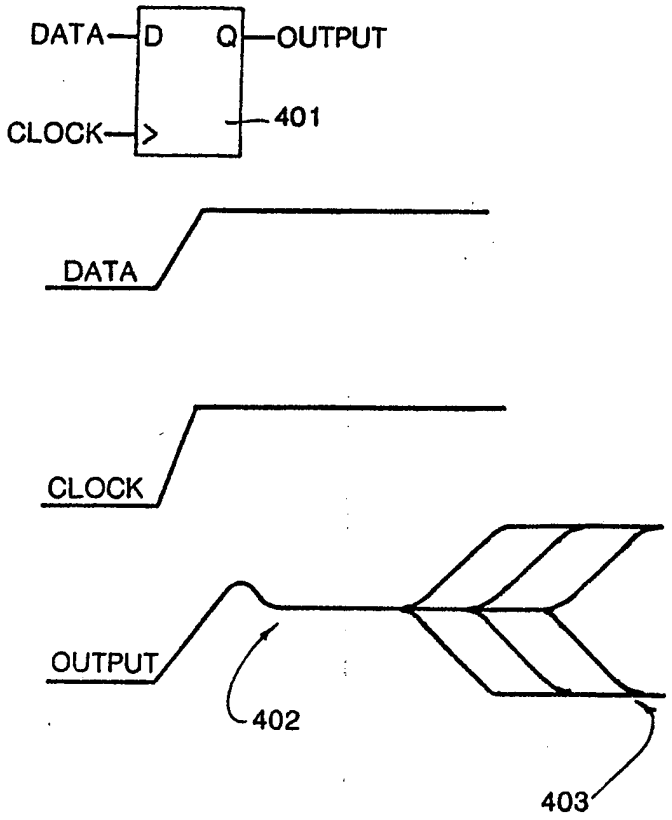


FIG 4

BINARY SIGNAL

MULTIVALUED SIGNALS

D_1, D_2 ARBITRARY DATA

SIGNALS WHICH MAY BE "METASTABLE"

FURRY WAVEFORMS

$\overline{DAV}_A$, $\overline{RFD}_B$ NEGATIVE LOGIC

* A METASTABLE SAMPLE SETTING TO THE NEW VALUE

** SETTING TO THE OLD VALUE

FIG 5

0217486

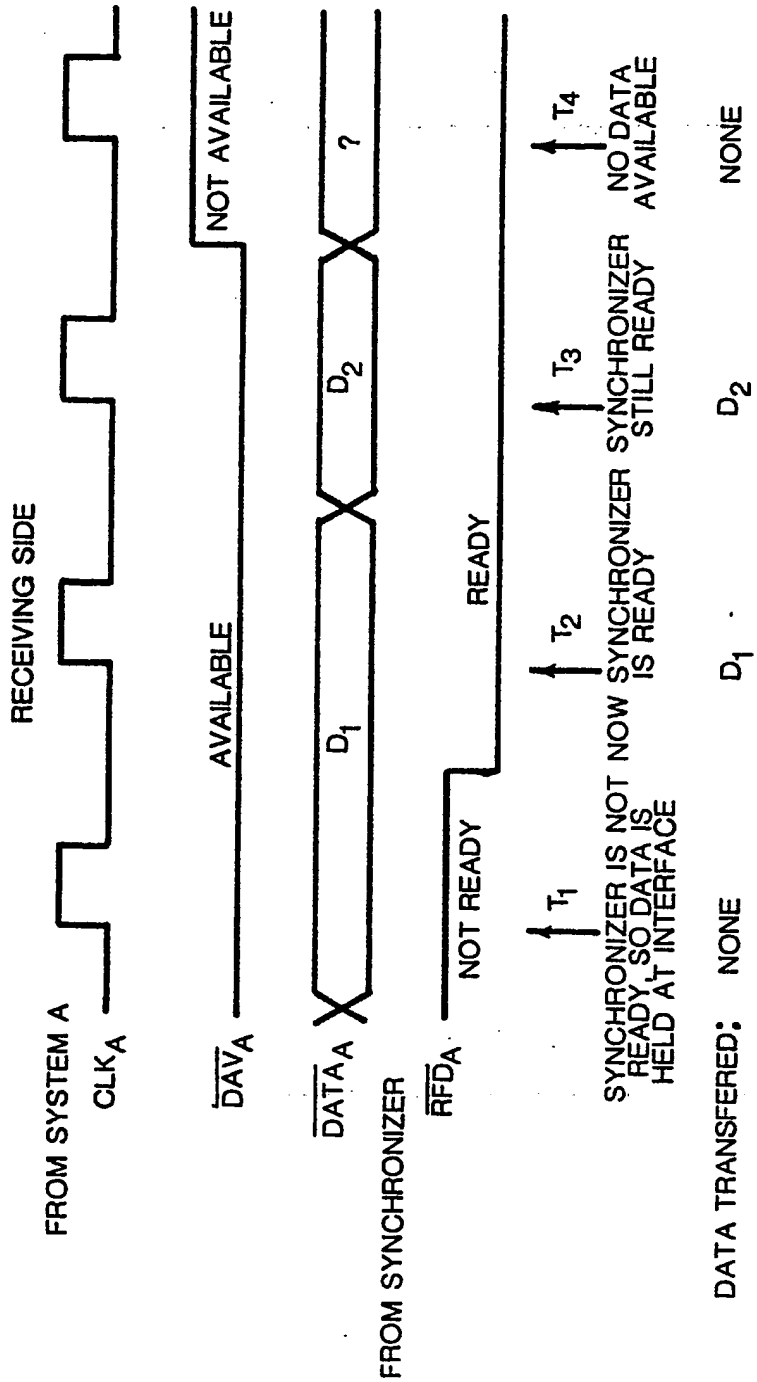

RECEIVING SIDE
FROM SYSTEM A
CLK$_A$
AVAILABLE
NOT AVAILABLE
$\overline{DAV}_A$
$\overline{DATA}_A$
D_1
D_2
?
FROM SYNCHRONIZER
$\overline{RFD}_A$
NOT READY
READY
T_1
T_2
T_3
T_4
SYNCHRONIZER IS NOT READY, SO DATA IS HELD AT INTERFACE
NOW SYNCHRONIZER IS READY
SYNCHRONIZER STILL READY
NO DATA AVAILABLE
DATA TRANSFERED: NONE
D_1
D_2
NONE

FIG 6

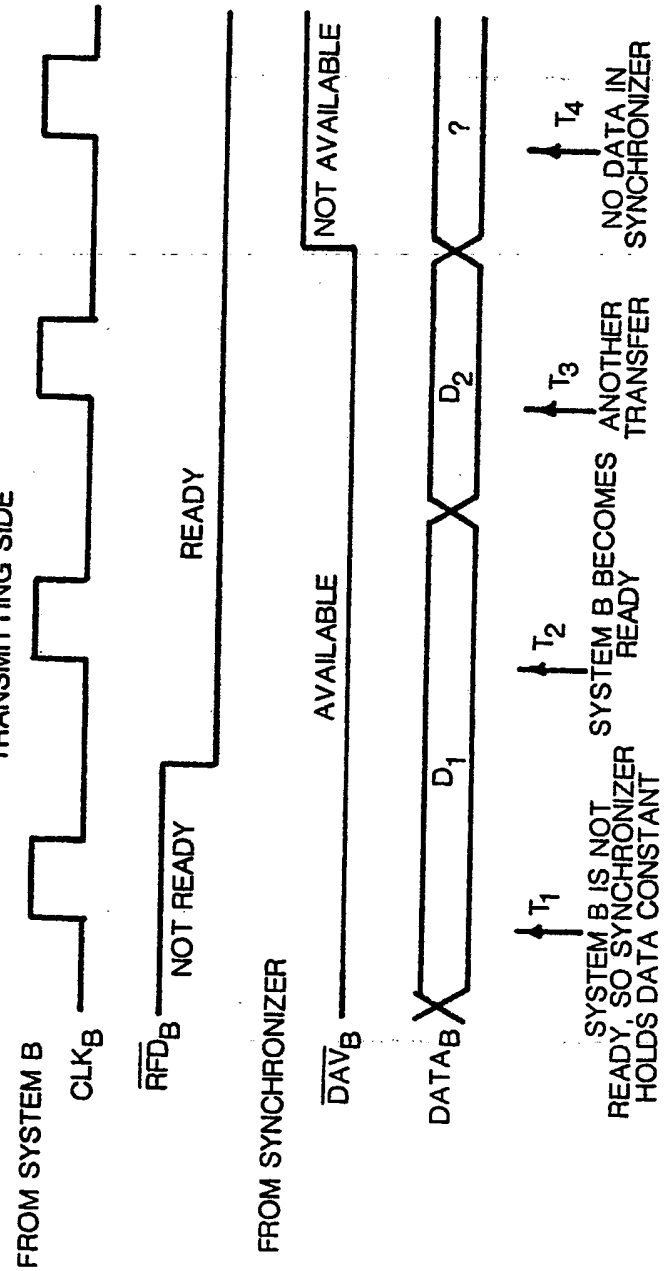

TRANSMITTING SIDE
FROM SYSTEM B
CLK_B
RFD_B
NOT READY
READY
FROM SYNCHRONIZER
DAV_B
AVAILABLE
NOT AVAILABLE
DATA_B
D1
D2
?
T1
SYSTEM B IS NOT READY, SO SYNCHRONIZER HOLDS DATA CONSTANT
T2
SYSTEM B BECOMES READY
T3
ANOTHER TRANSFER
T4
NO DATA IN SYNCHRONIZER
DATA TRANSFERED: NONE
D1
D2
NONE

FIG 7

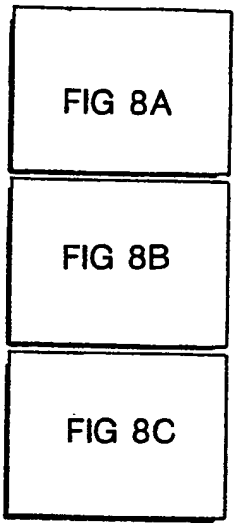


FIG 8

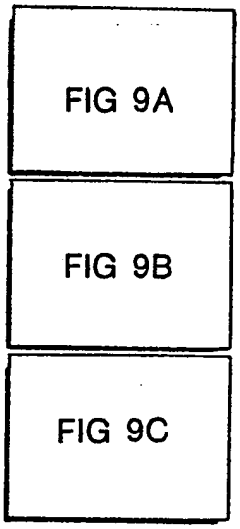


FIG 9

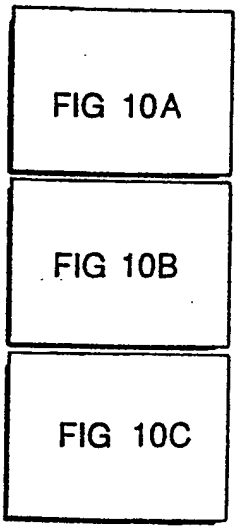


FIG 10

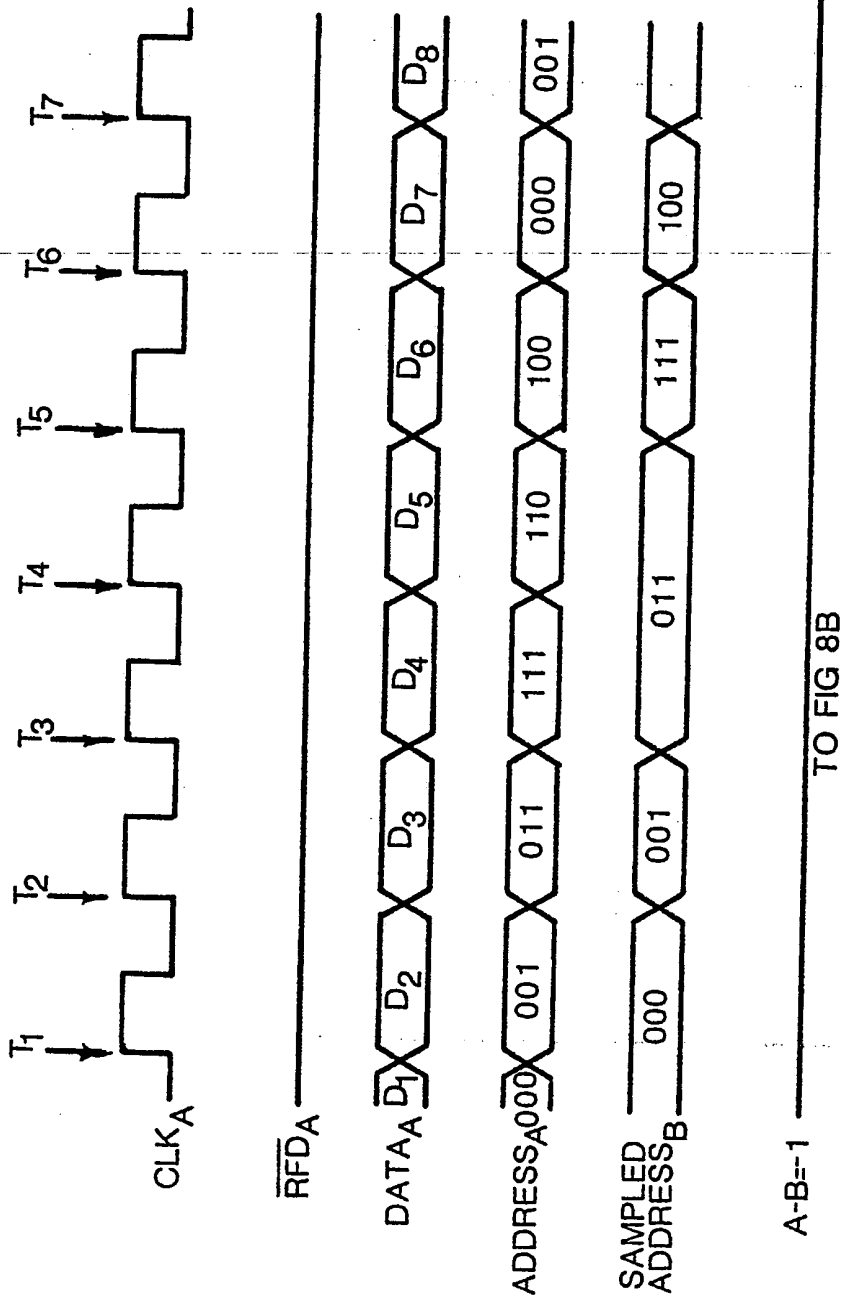

T1
T2
T3
T4
T5
T6
T7
CLKA
RFDA
DATAA
D1
D2
D3
D4
D5
D6
D7
D8
ADDRESSA
000
001
011
111
110
100
000
001
SAMPLED ADDRESSB
000
001
011
111
100
A-B=-1
TO FIG 8B

FIG 8A

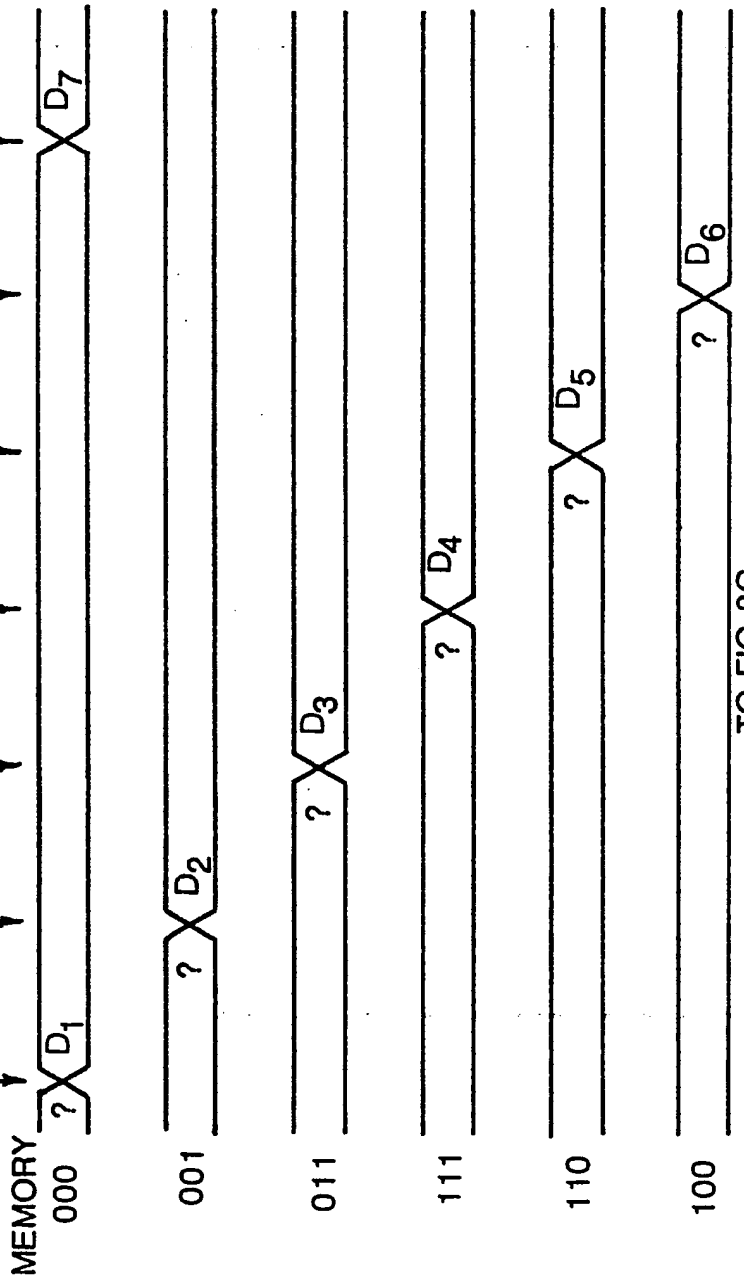

T1
T2
T3
T4
T5
T6
T7
MEMORY
000
001
011
111
110
100
?
D1
D2
D3
D4
D5
D6
D7
TO FIG 8C

FIG 8B

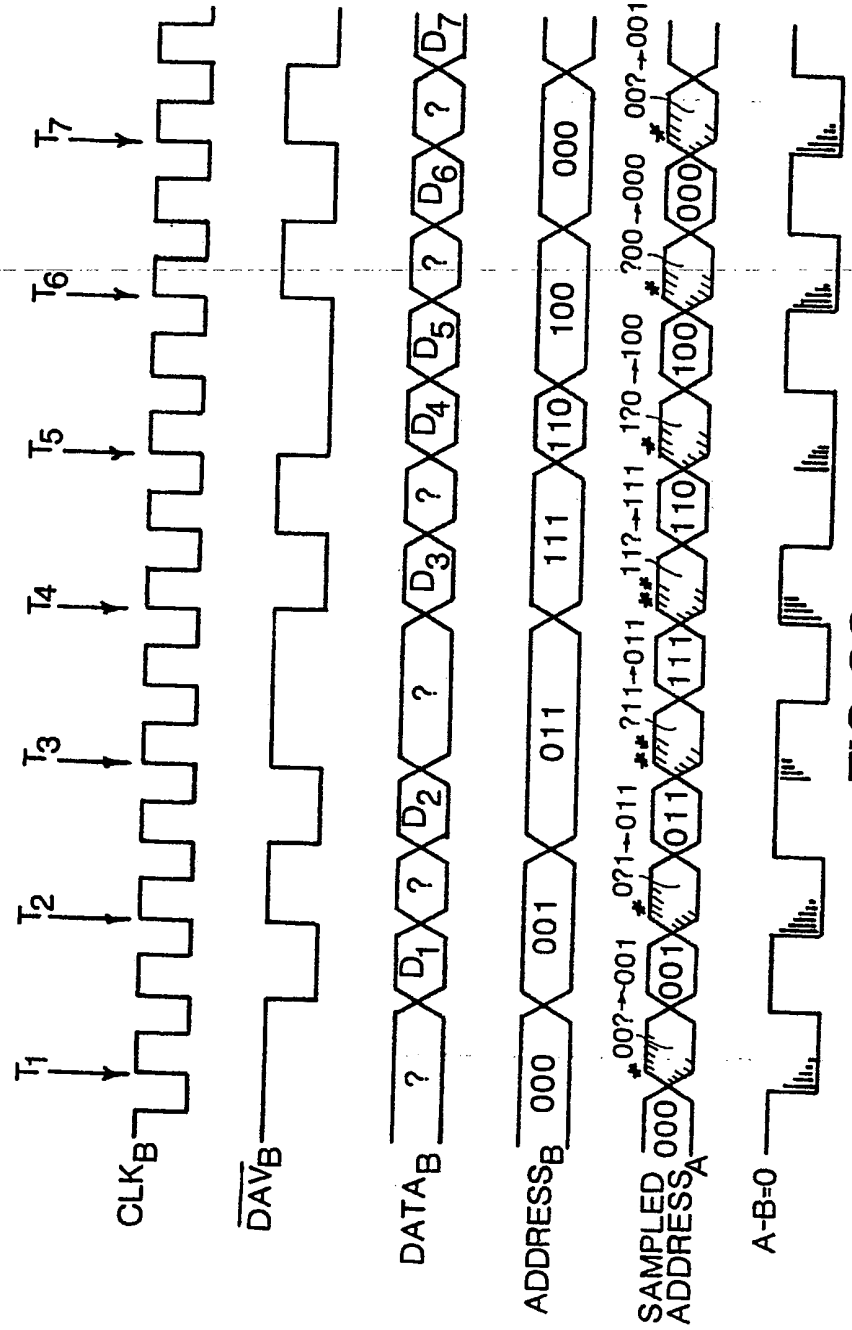

T1
T2
T3
T4
T5
T6
T7
CLKB
DAVB
DATAB
ADDRESSB
SAMPLED ADDRESSA
A-B=0
?
D1
D2
D3
D4
D5
D6
D7
000
001
011
111
110
100
00?→001
0?1→011
?11→011
11?→111
1?0→100
?00→000
00?→001

FIG 8C

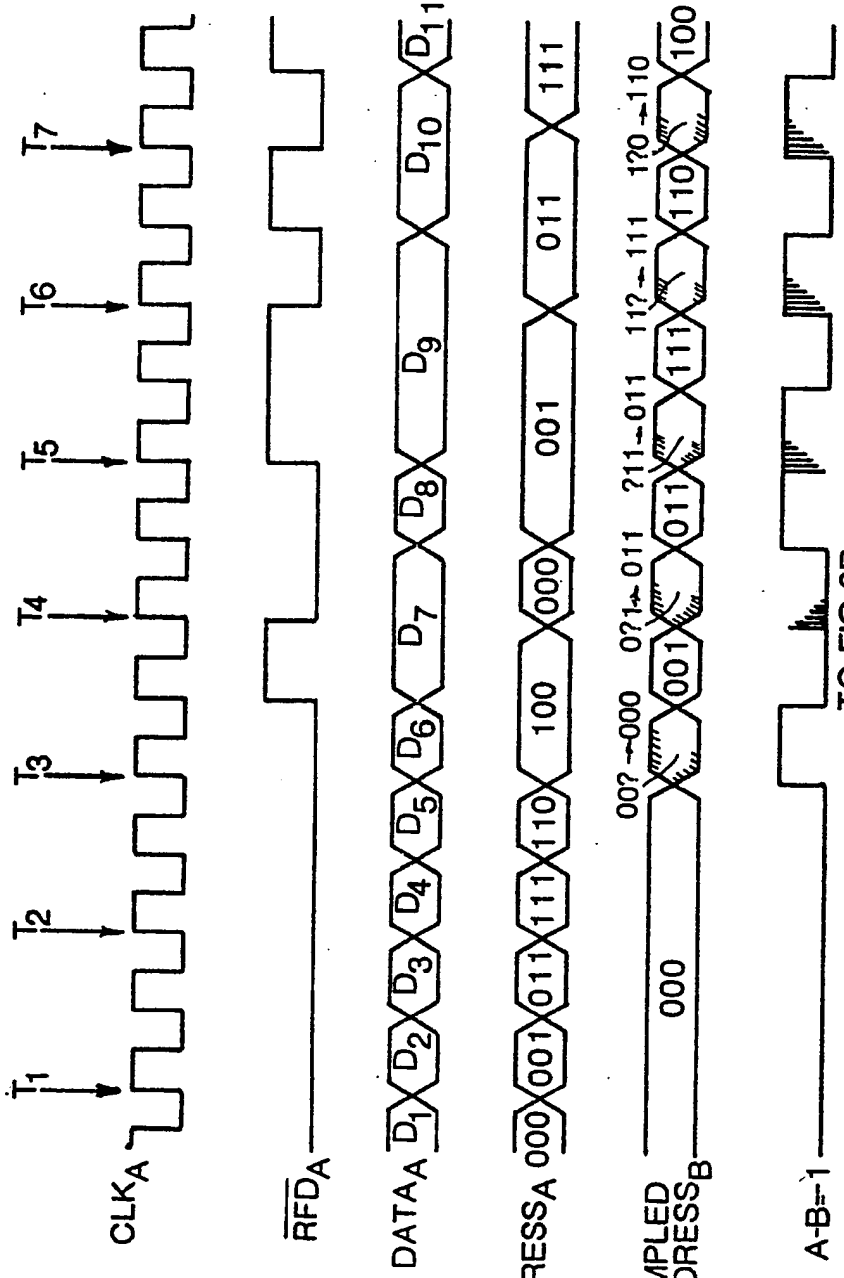

T1
T2
T3
T4
T5
T6
T7
CLKA
RFDA
DATAA
D1
D2
D3
D4
D5
D6
D7
D8
D9
D10
D11
ADDRESSA
000
001
011
111
110
100
000
001
011
111
SAMPLED ADDRESSB
000
00?→000
0?1→011
?11→011
11?→111
1?0→110
100
110
111
011
001
000
A-B=1
TO FIG 9B

FIG 9A

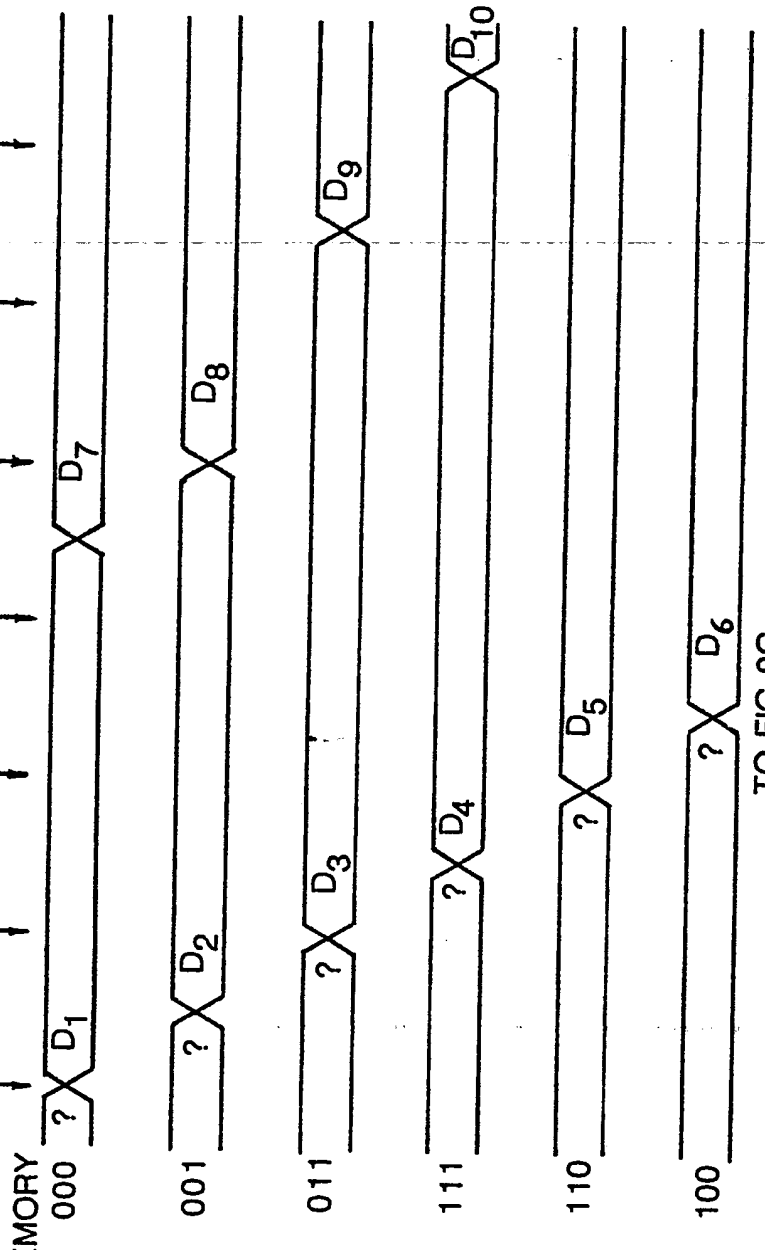

T1
T2
T3
T4
T5
T6
T7
MEMORY
000
001
011
111
110
100
?
D1
D2
D3
D4
D5
D6
D7
D8
D9
D10
TO FIG 9C

FIG 9B

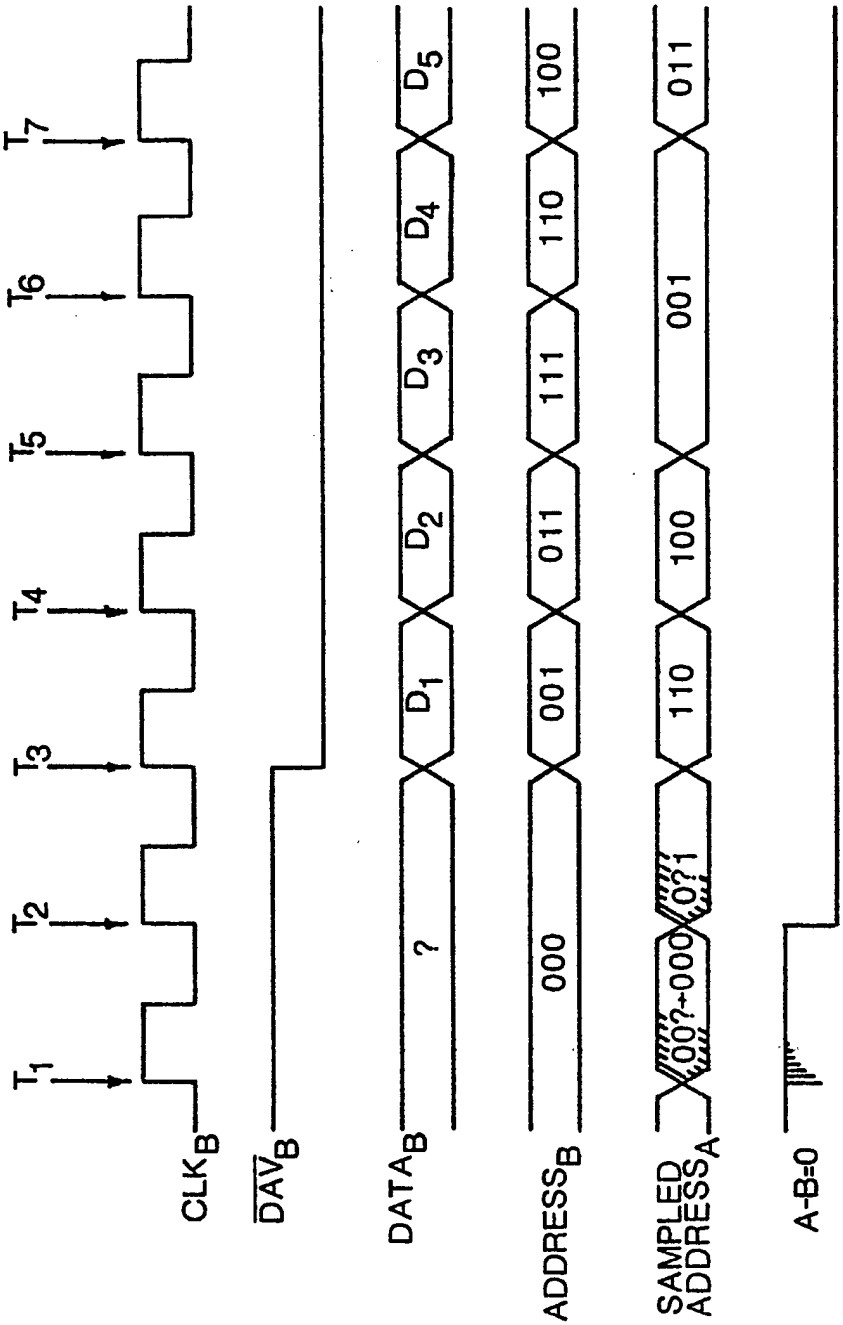

T1
T2
T3
T4
T5
T6
T7
CLKB
DAVB
DATAB
D1
D2
D3
D4
D5
?
ADDRESSB
000
001
011
111
110
100
SAMPLED ADDRESSA
000-001
110
100
001
011
A-B=0

FIG 9C

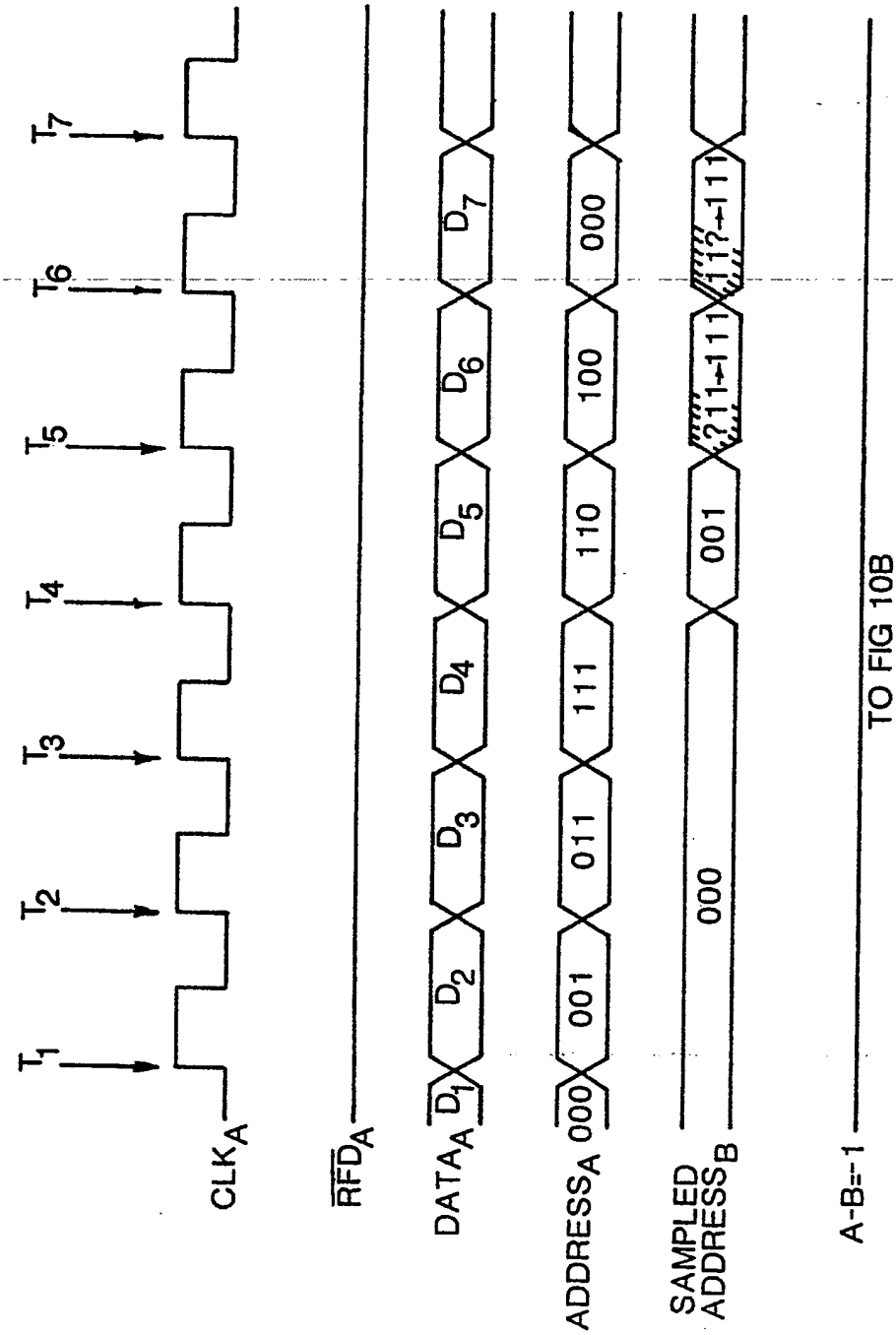
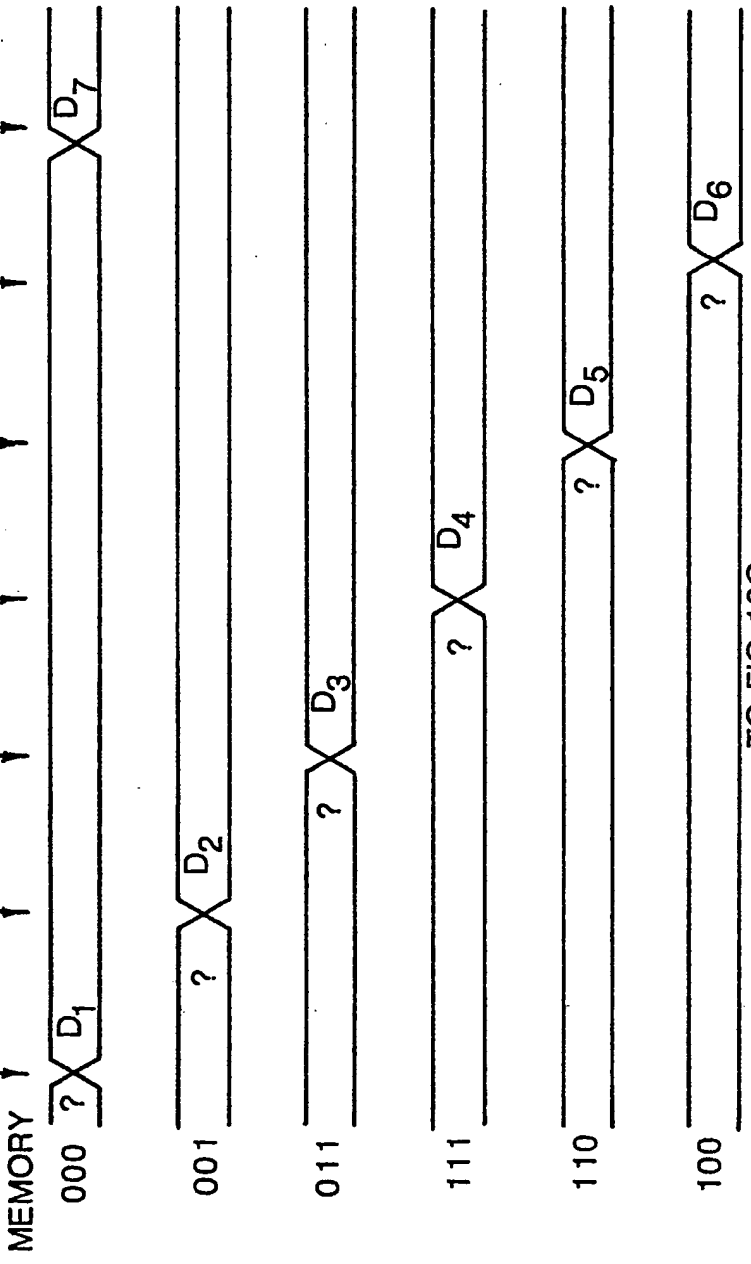

T1
T2
T3
T4
T5
T6
T7
CLKA
RFDA
DATAA
D1
D2
D3
D4
D5
D6
D7
ADDRESSA
000
001
011
111
110
100
000
SAMPLED ADDRESSB
000
001
211-111
111-111
A-B=-1
TO FIG 10B

FIG 10A



TO FIG 10C

FIG 10B

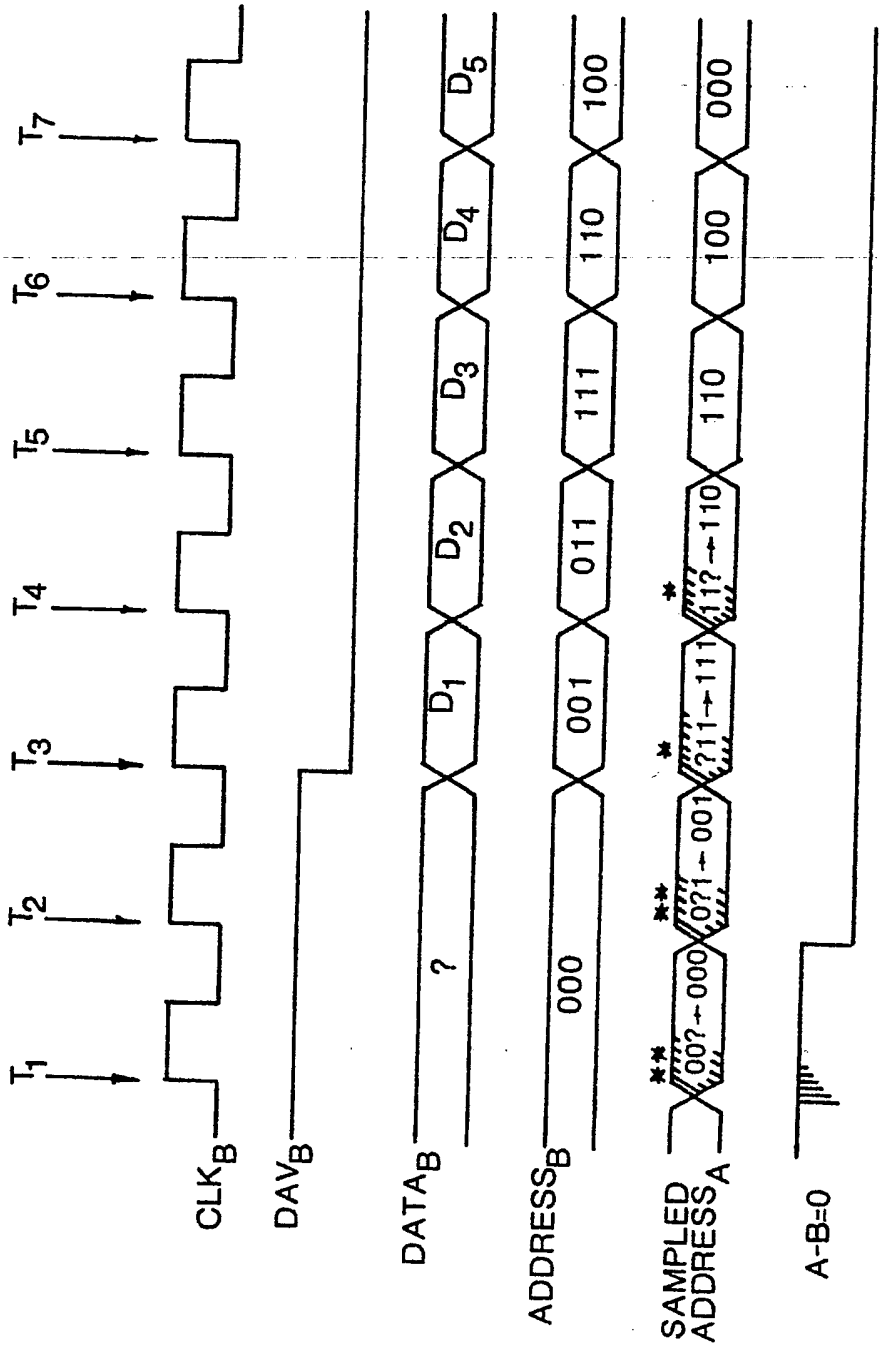

T1
T2
T3
T4
T5
T6
T7
CLKB
DAVB
DATAB
?
D1
D2
D3
D4
D5
ADDRESSB
000
001
011
111
110
100
SAMPLED ADDRESSA
002←000
021←001
211→111
112→110
110
100
000
A-B=0

FIG 10C